(19) JAPANESE PATENT OFFICE

PATENT ABSTRACTS OF JAPAN

(11) Publication number: 10333929 A

(43) Date of publication of application: 18 . 12 . 98

(51) Int. Cl G06F 9/46
G06F 15/16

(21) Application number: 10070882

(22) Date of filing: 19 . 03 . 98

(30) Priority: 31 . 03 . 97 JP 09 79261

(71) Applicant: FUJITSU LTD

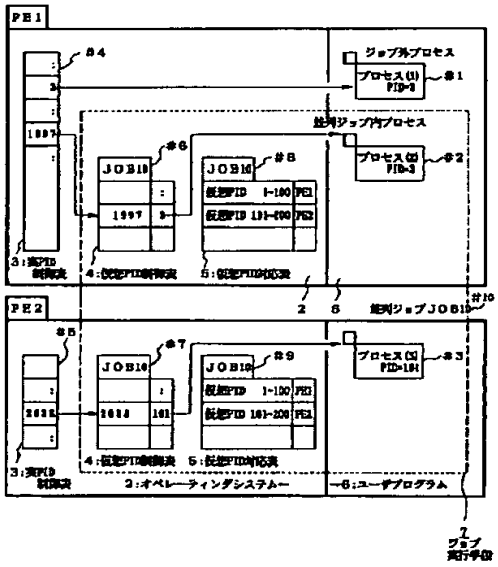
(72) Inventor: WAKABAYASHI HIROHIKO

(54) JOB EXECUTION SYSTEM AND RECORDING MEDIUM

(57) Abstract:

PROBLEM TO BE SOLVED: To prevent the collision of processes by registering virtual processes respectively allocated to the plural processes in a virtual PID control table, allocating the empty real PID of a real PID control table, registering it in correspondence and making the respective processes communicate with each other by using the virtual process and perform processings within a job.

SOLUTION: Plural processors PE1 and PE2 are constituted of the real PID control table 3 and a job execution means 7, etc. Further, the job execution means 7 provides the virtual PID control table 4 for the respective jobs and registers unique virtual process IDs respectively allocated to the plural processes for executing the jobs in the virtual PID control table 4. Then, the empty real PID of the real PID control table 3 is allocated and registered in correspondence and the respective processes communicate with each other by using the virtual process IDs and execute the processing. Thus, the collision between the process IDs is eliminated without the reservation of the process ID.



(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号

特開平10－333929

(43)公開日　平成10年(1998)12月18日

(51)Int.Cl.[6]		識別記号	FI		
Ｇ０６Ｆ	9/46	３６０	Ｇ０６Ｆ	9/46	３６０Ｂ
	15/16			15/16	４２０Ｊ

審査請求　未請求　請求項の数５　ＯＬ　（全９頁）

(21)出願番号　特願平10－70882

(22)出願日　平成10年(1998)３月19日

(31)優先権主張番号　特願平9－79261

(32)優先日　平９(1997)３月31日

(33)優先権主張国　日本（ＪＰ）

(71)出願人　000005223
富士通株式会社
神奈川県川崎市中原区上小田中４丁目１番１号

(72)発明者　若林　裕彦
神奈川県川崎市中原区上小田中４丁目１番１号　富士通株式会社内

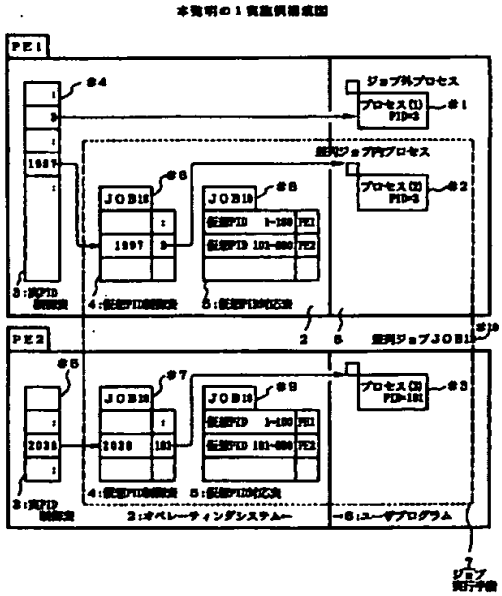
(74)代理人　弁理士　岡田　守弘

(54)【発明の名称】　ジョブ実行システムおよび記録媒体

(57)【要約】

【課題】　本発明は、ジョブ実行システムおよび記録媒体に関し、ジョブ毎に任意の一意の仮想プロセスＩＤの割り当てを実現し、凍結したジョブを解凍したときにプロセスＩＤの予約なしにプロセスＩＤの衝突を無くして即時にジョブ内のプロセス間でデータのやりとりを行う処理の再開を可能にすることを目的とする。

【解決手段】　複数のプロセッサ上でそれぞれ処理を実行するプロセスにそれぞれ割り当てた一意の実プロセスＩＤを登録して管理する実ＰＩＤ制御表と、複数のプロセッサ上でジョブを実行する複数のプロセスに対してジョブ内でそれぞれ割り当てた一意の仮想プロセスＩＤを、一意の実プロセスＩＤにそれぞれ対応づけて登録して管理する仮想ＰＩＤ制御表とを各プロセッサに設け、ジョブ内では各プロセスが仮想プロセスＩＤを用いて相互に通信し、処理を実行する手段を備えるように構成する。



1

【特許請求の範囲】

【請求項１】複数プロセッサ上で任意数のプロセスによってジョブを実行するジョブ実行システムにおいて、
複数のプロセッサ上でそれぞれ処理を実行するプロセスにそれぞれ割り当てた一意の実プロセスＩＤを登録して管理する実ＰＩＤ制御表と、
複数のプロセッサ上でジョブを実行する複数のプロセスに対して当該ジョブ内でそれぞれ割り当てた一意の仮想プロセスＩＤを、上記一意の実プロセスＩＤにそれぞれ対応づけて登録して管理する仮想ＰＩＤ制御表とを各プロセッサに設け、
上記ジョブ内では各プロセスが仮想プロセスＩＤを用いて相互に通信し、処理を実行する手段を備えたことを特徴とするジョブ実行システム。

【請求項２】各プロセッサ内にジョブ毎に仮想プロセスＩＤがいずれのプロセッサに存在するかを登録する仮想ＰＩＤ対応表を設け、
ジョブ内の仮想プロセスＩＤがいずれのプロセッサに存在するかを判別してそのプロセッサにデータを送信してジョブ内の各プロセスが相互にデータの送受信を行うことを特徴とする請求項１記載のジョブ実行システム。

【請求項３】ジョブの凍結指示に対応して、指示されたジョブの上記仮想ＰＩＤ制御表とジョブ内のプロセス、あるいは更に上記仮想ＰＩＤ対応表を退避して保存することを特徴とする請求項１あるいは請求項２記載のジョブ実行システム。

【請求項４】ジョブの解凍指示に対応して、上記保存しておいたジョブの仮想ＰＩＤ制御表とジョブ内のプロセス、あるいは更に上記仮想ＰＩＤ対応表を復元した後、当該仮想ＰＩＤ制御表中に登録されている仮想プロセスＩＤに対して上記実ＰＩＤ制御表の空の実プロセスＩＤを割り当てて登録した後、ジョブ内の各プロセスが処理を再開することを特徴とする請求項１ないし請求項３記載のいずれかのジョブ実行システム。

【請求項５】複数のプロセッサ上でそれぞれ処理を実行するプロセスにそれぞれ割り当てた一意の実プロセスＩＤを実ＰＩＤ制御表に登録して管理する手段と、
複数のプロセッサ上でジョブを実行する複数のプロセスに対して当該ジョブ内でそれぞれ割り当てた一意の仮想プロセスＩＤを、上記一意の実プロセスＩＤにそれぞれ対応づけて仮想ＰＩＤ制御表に登録して管理する手段と、
上記ジョブ内では各プロセスが仮想プロセスＩＤを用いて相互に通信し、処理を実行する手段と、して機能させるプログラムを記録したコンピュータ読取可能な記録媒体。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、複数プロセッサ上で任意数のプロセスによってジョブを実行するジョブ実行システムおよび記録媒体に関するものである。

2

【０００２】

【従来の技術】従来、複数のＰＥ（プロセッサ）間にまたがるプロセスの集合体によって実行するいわゆる並列ジョブを任意の時点で中断させ、その状態をファイルなどに保存して凍結し、システムのＩＰＬなどを経た後に、再びファイルから凍結したジョブを読み出し、当該ジョブの実行を継続するジョブ凍結・解凍機能がある。

【０００３】このジョブ凍結・解凍の際に問題となるのは次の点である。ジョブに属するプロセス中では、そのプロセスの識別子（プロセスＩＤ）を使用して、ジョブ内のプロセス間のデータのやりとりを行うが、ジョブ凍結の際には、プログラム空間内にそのプロセスＩＤの情報が残っているため、ジョブ解凍時には、ジョブ内のプロセスについては、以前に持っていたプロセスＩＤを保証した形で、ジョブを再現してプロセスＩＤの衝突を回避する必要がある。

【０００４】このとき、当該ジョブ外のプロセスで、そのプロセスＩＤと同一のプロセスＩＤが既に使用中である場合には、そのプロセスの終了を待たなければ、ジョブを再開できない。

【０００５】これを解決するため、プロセスの凍結時にプロセスＩＤに関する情報を格納しておき、システム再起動時に格納情報をもとにプロセスＩＤを予約し、ジョブを解凍して再現し、他のジョブが非所望に当該プロセスＩＤを使用することによる待ちを無くし、即時に予約しておいた以前のプロセスＩＤを用いてプロセス間のデータのやりとりを開始して処理を再開するようにしていた。

【０００６】

【発明が解決しようとする課題】上述したように、従来のジョブ凍結・解凍機能では、解凍時のプロセスＩＤの衝突を避けるために、システムの起動時にプロセスＩＤを予め予約しておく必要があり、このために解凍するジョブに応じた予約のための格納情報を確保し、システム起動時に予約して確保する必要が生じる。

【０００７】このような従来のジョブ凍結・解凍機能では、予めプロセスＩＤの予約が必要であり、凍結された任意のジョブを解凍できないという制約が生じる問題があった。

【０００８】また、並列プログラムでは、ジョブを構成するプログラムがＰＥの数だけ存在するため、多数のプロセスＩＤを予約しておく必要があり、その分だけシステム中で常に使用できないプロセスＩＤが存在し、他のプロセスのプロセスＩＤの採番範囲を圧迫してしまうという問題も発生する。特にジョブ凍結・解凍機能が要求される大規模なジョブ程、並列度が高く、多数のプロセスを必要とする傾向があり、極めて多数のプロセスＩＤが使用できなくなってしまうという問題がある。

【０００９】本発明は、これらの問題を解決するため、

一意な実プロセスＩＤを管理する実ＰＩＤ制御表の他にジョブ内の仮想プロセスＩＤを一意に管理する仮想ＰＩＤ制御表を設けて実プロセスＩＤと仮想プロセスＩＤとの対応づけを行い、ジョブ毎に任意の一意の仮想プロセスＩＤの割り当てを実現し、凍結したジョブを解凍したときにプロセスＩＤの予約なしにプロセスＩＤの衝突を無くして即時にジョブ内のプロセス間でデータのやりとりを行う処理の再開を可能にすることを目的としている。

【００１０】

【課題を解決するための手段】図１を参照して課題を解決するための手段を説明する。図１において、プロセッサＰＥ１、ＰＥ２は、複数のプロセッサであって、プロセスを生成してジョブを実行するためのものであり、ここでは、実ＰＩＤ制御表３、およびジョブ実行手段７などから構成されるものである。

【００１１】実ＰＩＤ制御表３は、プロセスに一意に割り当てた実プロセスＩＤを登録して管理するものである。ジョブ実行手段７は、複数プロセッサ上で任意個のプロセスによってジョブを実行するものであって、図示のように、仮想ＰＩＤ制御表４、仮想ＰＩＤ対応表５、および複数のプロセスから構成されるものである。

【００１２】仮想ＰＩＤ制御表４は、ジョブ内のプロセスに一意に割り当てた仮想プロセスＩＤに対応づけて実プロセスＩＤを登録して管理するものである。仮想ＰＩＤ対応表５は、仮想プロセスＩＤがいずれのプロセッサにあるかを登録したものである。

【００１３】次に、動作を説明する。ジョブ毎に仮想ＰＩＤ制御表４を設けてジョブを実行する複数のプロセスにそれぞれ割り当てた一意の仮想プロセスＩＤを当該仮想ＰＩＤ制御表４に登録すると共に実ＰＩＤ制御表３の空の実ＰＩＤを割り当てて対応づけて登録し、ジョブ内では各プロセスが仮想プロセスＩＤを用いて相互に通信し、処理を実行するようにしている。

【００１４】この際、各プロセッサ内に仮想プロセスＩＤがいずれのプロセッサに存在するかを登録する仮想ＰＩＤ対応表５を設け、ジョブ内の仮想プロセスＩＤがいずれのプロセッサに存在するかを判別してそのプロセッサにデータを送信してジョブ内の各プロセスが相互にデータの送受信を行い、処理を実行するようにしている。

【００１５】また、ジョブの凍結指示に対応して、指示されたジョブの仮想ＰＩＤ制御表４とジョブ内のプロセス、あるいは更に仮想ＰＩＤ対応表５を退避して保存するようにしている。

【００１６】また、ジョブの解凍指示に対応して、保存しておいたジョブの仮想ＰＩＤ制御表４とジョブ内のプロセス、あるいは更に仮想ＰＩＤ対応表５を復元した後、仮想ＰＩＤ制御表４中に登録されている仮想プロセスＩＤに対して実ＰＩＤ制御表３の空の実プロセスＩＤを割り当てて登録した後、ジョブ内の各プロセスが処理を再開するようにしている。

【００１７】従って、一意な実プロセスＩＤを管理する実ＰＩＤ制御表３の他にジョブ内の仮想プロセスＩＤを一意に管理する仮想ＰＩＤ制御表４を設けて実プロセスＩＤと仮想プロセスＩＤとの対応づけを行うことにより、ジョブ毎に任意の一意の仮想プロセスＩＤの割り当てを実現し、凍結したジョブを解凍したときにプロセスＩＤの予約なしにプロセスＩＤの衝突を無くして即時にジョブ内のプロセス間でデータのやりとりを行う処理を再開することが可能となる。

【００１８】

【発明の実施の形態】次に、図１から図５を用いて本発明の実施の形態および動作を順次詳細に説明する。ここで、記録媒体から読み出したプログラムあるいは外部記憶装置であるハードディスク装置などから読み出したプログラム、またはセンタの外部記憶装置から読み出して回線を介して転送を受けたプログラムを主記憶にローディングして起動し、以下に説明する各種処理を行うようにしている。

【００１９】図１は、本発明の１実施例構成図を示す。図１において、ＰＥ１、ＰＥ２は、複数のプロセッサのうちの２つを表したものであって、図示のように、オペレーティングシステム２およびユーザプログラム６の部分から構成されるものである。

【００２０】オペレーティングシステム（ＯＳ）２は、全体を統括制御するものであって、ここでは、図示のように、実プロセスＩＤ（実ＰＩＤ）および仮想プロセスＩＤ（仮想ＰＩＤ）を割り当ておよび管理するための実ＰＩＤ制御表３およびジョブ実行手段７を構成するうちの仮想ＰＩＤ制御表４、仮想ＰＩＤ対応表５からなるものである。

【００２１】実ＰＩＤ制御表３は、ユーザプログラム６の部分に創成したプロセスに一意に割り当て実プロセスＩＤ（実ＰＩＤ）を登録して管理するものである。仮想ＰＩＤ制御表４は、ジョブ内でユーザプログラム６の部分に創成したプロセスに一意に割り当て仮想プロセスＩＤ（仮想ＰＩＤ）を登録して管理するものである。

【００２２】仮想ＰＩＤ対応表５は、仮想プロセスＩＤの所定範囲がいずれのプロセッサ（ＰＥ）にあるかを登録したものである（図５参照）。プロセスは、ユーザプログラム６の部分に創成して各種処理を実行するものであって、相互にデータを通信などするために一意のプロセスＩＤ（実ＰＩＤあるいは仮想ＰＩＤ）を割り当てたものである。

【００２３】次に、図２のフローチャートに示す順序に従い、図１の構成のもとでプロセスＩＤ（実ＰＩＤおよび仮想ＰＩＤ）を割り当てるときの手順を詳細に説明する。図２は、本発明の動作説明フローチャートを示す。ここで、ＰＥ１、ＰＥ２は、図１のプロセッサであるＰＥ１、ＰＥ２である。

5

【００２４】図2において、Ｓ1、Ｓ2は、ＰＥ1、ＰＥ2がＯＳのＩＰＬを行う。ここでは、ＯＳのＩＰＬの過程でＰＥ1、ＰＥ2は、プロセスの実プロセスＩＤを一意に割り当てて登録して管理するための実ＰＩＤ制御表＃４、＃５の作成を行う。

【００２５】Ｓ3は、ＰＥ1に並列ジョブ＃１０が投入される。この延長で、ＰＥ1、ＰＥ2を使用する並列ジョブ＃１０が実行されることとなる。Ｓ4、Ｓ5は、Ｓ3で並列ジョブ＃１０が投入されると、ジョブ毎の仮想ＰＩＤ制御表＃６、＃７の作成を行う。これにより、ジョブ毎に仮想プロセスＩＤと実プロセスＩＤの対応関係を登録して管理する仮想ＰＩＤ制御表＃６、＃７の作成が行われたこととなる。

【００２６】Ｓ6、Ｓ7は、マスタのプロセッサＰＥ1が仮想ＰＩＤ対応表＃８を作成して各ＰＥ、ここではＰＥ2への配布を行い当該ＰＥ2で仮想ＰＩＤ対応表＃９を作成する。この仮想ＰＩＤ対応表＃８、＃９により、各ＰＥにおいて仮想プロセスＩＤとＰＥとの対応関係を認識し、仮想プロセスＩＤをもとにいずれのＰＥにデータを送信すればよいかを判別することが可能となる。ここでは、仮想ＰＩＤ対応表＃８、＃９には

仮想プロセスＩＤ	ＰＥ
００１～１００	ＰＥ1
１０１～２００	ＰＥ2

と登録されているので、仮想プロセスＩＤが００１～１００まではＰＥ1に存在し、１０１～２００まではＰＥ2に存在することが判明する

Ｓ8は、ＰＥ1で親プロセス＃２を生成する。そして、親プロセス＃２は、Ｓ9で子プロセス＃３をＰＥ2に生成する。

【００２７】Ｓ10は、Ｓ8で親プロセスを生成したことに対応して、ＰＥ1では、実ＰＩＤ制御表＃４の空の実プロセスＩＤとして「１９９７」を割り当てて登録する。Ｓ11は、Ｓ10と同様に、Ｓ9で子プロセスを生成したことに対応して、ＰＥ2では、実ＰＩＤ制御表＃５の空の実プロセスＩＤとして「２０３８」を割り当てて登録する。

【００２８】Ｓ12は、空き仮想プロセスＩＤ（３）の割り当てを行うと共に、実プロセスＩＤ（１９９７）と仮想プロセスＩＤ（３）を対応づけて仮想ＰＩＤ制御表＃６への登録を行う。これにより、仮想ＰＩＤ制御表＃６には、実プロセスＩＤ（３）と仮想プロセスＩＤ（１９９７）との対応関係が登録されたこととなる。

【００２９】Ｓ13は、Ｓ12と同様に、仮想プロセスＩＤ（１０１）の割り当てを行うと共に、実プロセスＩＤ（２０３８）と仮想プロセスＩＤ（１０１）を対応づけて仮想ＰＩＤ制御表＃７への登録を行う。これにより、仮想ＰＩＤ制御表＃７には、実プロセスＩＤ（２０３８）と仮想プロセスＩＤ（１０１）との対応関係が登録されたこととなる。

6

【００３０】以上の手順によって、実プロセスＩＤと仮想プロセスＩＤを制御するための内部データ構造が構築されたこととなる。そして、ユーザプログラムへは、Ｓ12、Ｓ13で割り当てられた仮想プロセスＩＤが通知され、ユーザプログラムは自分、あるいは親プロセス＃２、子プロセス＃３のプロセスＩＤを認識する。ここでは、親プロセス＃２は「3」（実際は実プロセスＩＤ、１９９７）、子プロセス＃３は「１０１」（実際は実プロセスＩＤ、２０３８）と認識する。

【００３１】次に、図3のフローチャートに示す順序に従い、図1の構成のもとで、図2の手順によって実プロセスＩＤと仮想プロセスＩＤを制御するための内部データ構造が構築された状態で、ジョブを実行するときの動作を詳細に説明する。

【００３２】図3は、本発明の動作説明フローチャートを示す。ここで、ＰＥ1、ＰＥ2は、図1のプロセッサであるＰＥ1、ＰＥ2である。図3において、Ｓ21は、プロセス＃２（親プロセス）からＰＩＤ＝１０１へデータを送る。

【００３３】Ｓ22は、ＰＥ1のオペレーティングシステム2が仮想ＰＩＤ対応表＃８をみて通信先ＰＥを決定する。ここで、ＰＩＤ＝１０１は、ＰＥ2と決定する。Ｓ23は、Ｓ22でＰＥ2と決定されたので、ＰＥ2へ並列ジョブ＃１０のＰＩＤ＝１０１のプロセスへデータを送る。これは、Ｓ22で通信先ＰＥがＰＥ2と決定されたので、ＰＥ1と同一ジョブ＃１０の配下の仮想プロセスＩＤ（１０１）のプロセスに対して、データを送信する。

【００３４】Ｓ24は、Ｓ23で送信されたデータを受信したＰＥ2のオペレーティングシステム2が、並列ジョブ＃１０の仮想ＰＩＤ制御表＃３を見てプロセス＃３へデータを渡す。そして、ジョブを実行する。

【００３５】以上によって、図1のＰＥ1の並列ジョブ＃１０内のプロセス＃２からＰＩＤ＝１０１へデータ送信したことに対応して、仮想ＰＩＤ対応表＃８を参照してＰＥ2にデータが渡され、ＰＥ2内で更に仮想ＰＩＤ制御表＃７を参照して並列ジョブ＃１１内のプロセス＃３にデータが渡されたこととなる。これらにより、仮想ＰＩＤ対応表を参照して該当するＰＥにデータを送信し、次に当該ＰＥ内で仮想ＰＩＤ制御表を参照して該当するプロセスにデータを渡し、ジョブ内で異なるＰＥ内のプロセスにデータを送信できたこととなる。

【００３６】次に、図4のフローチャートに示す順序に従い、図1の構成のもとで、ジョブの凍結および解凍について詳細に説明する。図4は、本発明の動作説明フローチャートを示す。

【００３７】図4において、Ｓ31は、ジョブ凍結指示を受信する。これは、ＯＳがシステム管理者あるいはユーザからジョブ凍結指示を受信する。そして、ジョブ実行中の各ＰＥにジョブ凍結指示を通知する。

【００３８】Ｓ３２は、ＰＥ２がＳ３１で通知したジョブ凍結指示を受信する。Ｓ３３、Ｓ３４は、各ＰＥ内で並列ジョブ＃１０の仮想ＰＩＤ対応表＃８、＃９、仮想ＰＩＤ制御表＃６、＃７とプロセス＃２、＃３をそれぞれファイルに退避して保存する。

【００３９】Ｓ３５は、ジョブ凍結完了する。以上によって、ジョブ凍結指示を受信したときに、凍結指示を受けたジョブを実行している各ＰＥは、当該ジョブの仮想ＰＩＤ対応表、仮想ＰＩＤ制御表、およびプロセスをそれぞれファイルに退避して保存する。

【００４０】Ｓ４０は、ジョブ解凍を開始する。Ｓ４１は、ジョブ解凍指示を受信する。これは、ＯＳがシステム管理者あるいはユーザからジョブ解凍指示を受信する。そして、ジョブを割り付ける各ＰＥにジョブ解凍指示を通知する。

【００４１】Ｓ４２は、ＰＥ２がＳ４１で通知したジョブ解凍指示を受信する。Ｓ４３、Ｓ４４は、各ＰＥ内で並列ジョブ＃１０の仮想ＰＩＤ対応表＃８、＃９、仮想ＰＩＤ制御表＃６、＃７の復元を行う。そして、更にプロセス＃２、＃３の情報も合わせてＰＥ１、ＰＥ２に復元する。

【００４２】Ｓ４５、Ｓ４６は、実ＰＩＤ制御表＃４、＃５の空プロセスＩＤの再割り当てと仮想ＰＩＤ制御表＃６、＃７への登録を行う。これにより、実ＰＩＤ制御表＃４、＃５の空の実プロセスＩＤを割り当てて、仮想ＰＩＤ制御表＃６、＃７に実プロセスＩＤと仮想プロセスＩＤとの対応づけが登録され、実プロセスＩＤと仮想プロセスＩＤとの対応関係が決定されたこととなる。

【００４３】Ｓ４７、Ｓ４８は、プロセス＃２、＃３の復元を行い、ジョブの実行を再開する。以上のように、仮想ＰＩＤ対応表、仮想ＰＩＤ制御表、プロセスの情報を復元した後、実ＰＩＤ制御表から空の実プロセスＩＤを割り当ててこの実プロセスＩＤを仮想ＰＩＤ制御表に登録して仮想プロセスＩＤとの対応関係を決定した後、プロセスを復元してジョブの実行を再開することが可能となる。このように、実プロセスＩＤの割り当てとして、実ＰＩＤ制御表を参照して空の実プロセスＩＤを仮想プロセスＩＤに対応づけて仮想ＰＩＤ制御表に登録しているため、確実に仮想プロセスＩＤに実プロセスＩＤを割り当てることができ、従来の他のプロセスが凍結時と同一の実プロセスＩＤを使用してそれの終了を待つまで待機するという事態の発生を無くし、迅速に実プロセスＩＤを仮想プロセスＩＤに割り当ててジョブの実行を再開することが可能となった。

【００４４】図５は、本発明の仮想ＰＩＤ対応表例を示す。この仮想ＰＩＤ対応表５は、仮想プロセスＩＤの範囲に対応づけてプロセッサＩＤを登録したものである。ここでは、仮想プロセスＩＤの範囲が００１～１００の場合に、プロセッサＰＥ１と登録し、仮想プロセスＩＤが００１～１００のプロセスは、プロセッサＰＥ１上で動作するものである旨が判明する。従って、仮想プロセスＩＤが００１～１００の範囲内の仮想プロセスＩＤへのデータ送信依頼の場合には、データをＰＥ１に送信すればよい。そして、ＰＥ１は、受信したデータについて、仮想ＰＩＤ制御表４を参照して該当する仮想プロセスＩＤのプロセスに渡す。

【００４５】

【発明の効果】以上説明したように、本発明によれば、一意な実プロセスＩＤを管理する実ＰＩＤ制御表３の他にジョブ内の仮想プロセスＩＤを一意に管理する仮想ＰＩＤ制御表４を設けて実プロセスＩＤと仮想プロセスＩＤとの対応づけを行う構成を採用しているため、ジョブ毎に任意の一意の仮想プロセスＩＤの割り当てを実現し、凍結したジョブを解凍したときに従来のプロセスＩＤの予約なしにプロセスＩＤの衝突を無くして即時にジョブ内のプロセス間でデータのやりとりを行う処理を再開することができる。また、ユーザプログラムの構成（プロセスの構成）を意識することなく、解凍時にプロセスＩＤの再割り当てを行うときに自動的にプロセスＩＤの衝突を回避してジョブの再開を確実かつ迅速に行い、処理の続行することが可能となる。

【図面の簡単な説明】

【図１】本発明の１実施例構成図である。

【図２】本発明の動作説明フローチャート（プロセスＩＤの割り当て）である。

【図３】本発明の動作説明フローチャート（仮想プロセスＩＤを使用したデータ送受信）である。

【図４】本発明の動作説明フローチャート（ジョブ凍結、ジョブ解凍時のプロセスＩＤの割り当て）である。

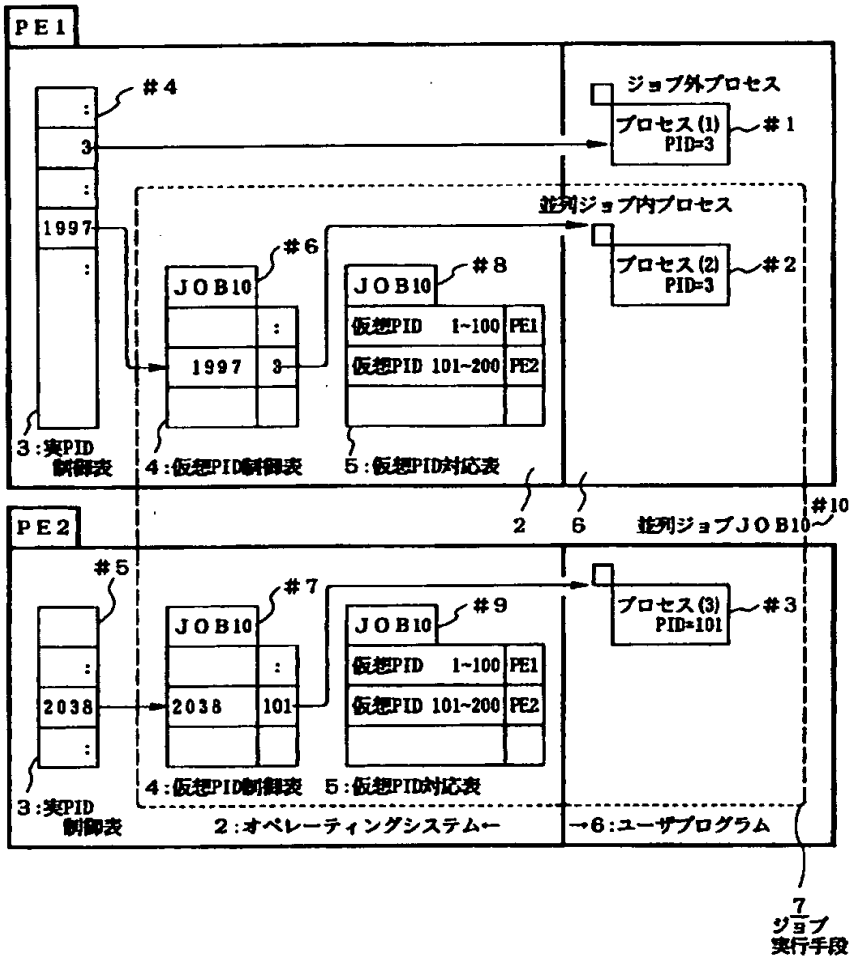
【図５】本発明の仮想ＰＩＤ対応表例である。

【符号の説明】

２：オペレーティングシステム（ＯＳ）
３：実ＰＩＤ制御表
４：仮想ＰＩＤ制御表
５：仮想ＰＩＤ対応表
６：ユーザプログラム
７：ジョブ実行手段
ＰＥ、ＰＥ１、ＰＥ２：プロセッサ

【図１】

本発明の１実施例構成図



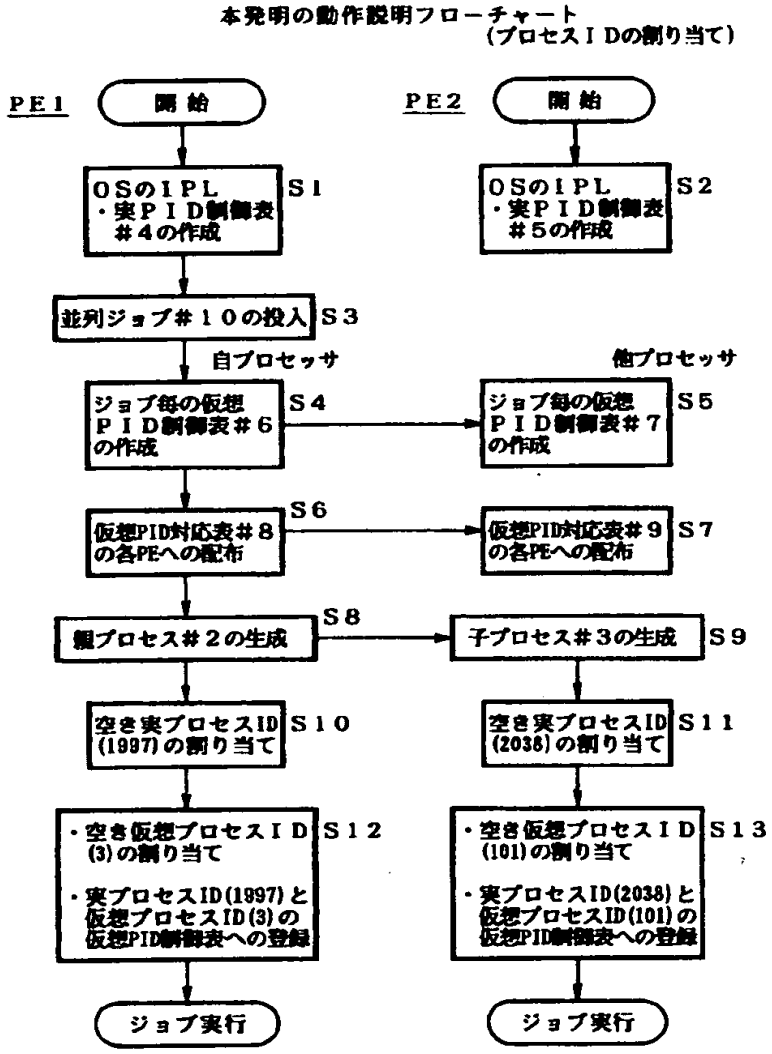
【図５】

本発明の仮想ＰＩＤ対応表例

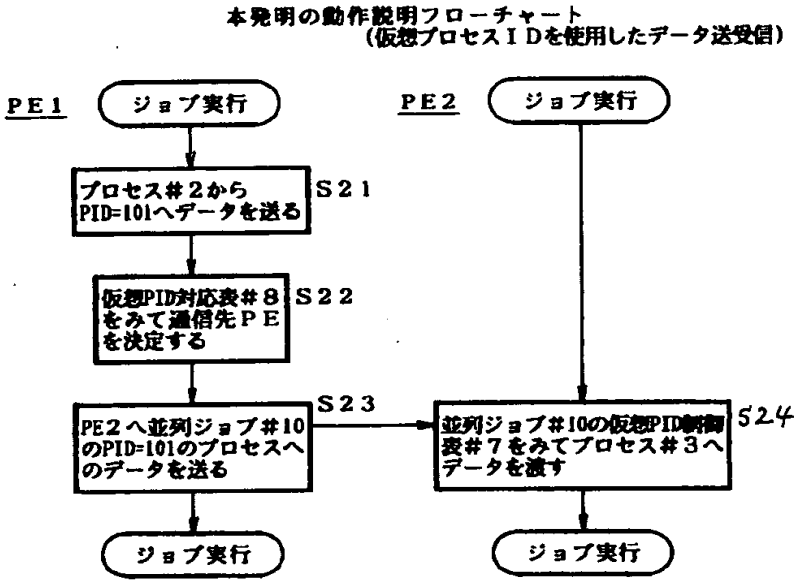
5

仮想プロセスＩＤ	プロセッサＩＤ	
001~100	PE1	
101~200	PE2	

【図２】


本発明の動作説明フローチャート
(プロセスＩＤの割り当て)
PE1
開始
OSのIPL
・実ＰＩＤ制御表
＃４の作成
S1
並列ジョブ＃１０の投入
S3
自プロセッサ
ジョブ毎の仮想
ＰＩＤ制御表＃６
の作成
S4
仮想PID対応表＃８
の各PEへの配布
S6
親プロセス＃２の生成
S8
空き実プロセスID
(1997)の割り当て
S10
・空き仮想プロセスＩＤ
(3)の割り当て
・実プロセスID(1997)と
仮想プロセスID(3)の
仮想PID制御表への登録
S12
ジョブ実行
PE2
開始
OSのIPL
・実ＰＩＤ制御表
＃５の作成
S2
他プロセッサ
ジョブ毎の仮想
ＰＩＤ制御表＃７
の作成
S5
仮想PID対応表＃９
の各PEへの配布
S7
子プロセス＃３の生成
S9
空き実プロセスID
(2038)の割り当て
S11
・空き仮想プロセスＩＤ
(101)の割り当て
・実プロセスID(2038)と
仮想プロセスID(101)の
仮想PID制御表への登録
S13
ジョブ実行

【図３】


本発明の動作説明フローチャート
(仮想プロセスＩＤを使用したデータ送受信)
PE1
ジョブ実行
プロセス＃２から
PID=101へデータを送る
S21
仮想PID対応表＃８
をみて通信先ＰＥ
を決定する
S22
PE2へ並列ジョブ＃10
のPID=101のプロセスへ
のデータを送る
S23
ジョブ実行
PE2
ジョブ実行
並列ジョブ＃10の仮想PID制御
表＃７をみてプロセス＃３へ
データを渡す
S24
ジョブ実行

【図４】

本発明の動作説明フローチャート
（ジョブ凍結、ジョブ解凍時のプロセスＩＤの割り当て）